

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : **09-046235**

(43)Date of publication of application : **14.02.1997**

(51)Int.Cl.

H03M 7/40

(21)Application number : **07-249956**

(71)Applicant : **NEC CORP**

(22)Date of filing : **27.09.1995**

(72)Inventor : **OKAMURA TOSHIHIKO**

(30)Priority

Priority number : **06239926**  
**07121135**

Priority date : **04.10.1994**  
**19.05.1995**

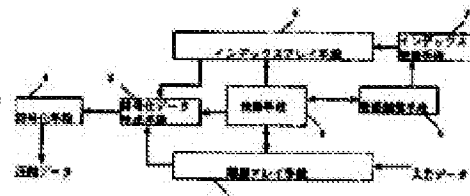
Priority country : **JP**  
**JP**

## (54) DATA COMPRESSION DEVICE

(57)Abstract:

PROBLEM TO BE SOLVED: To improve encoding speed by providing a retrieval means for retrieving the character string of a record array means and a registration control means controlling the registration of the character string in the record array means to the retrieval means.

SOLUTION: Character strings  $x(s)$ ,  $x(s+1)<$  are compared with the character string registered in the retrieval means 2 by using the retrieval means 2, and the longest matched string is obtained. An index supplied from an index update means 7 is distributed to an entry where  $x(s)$  is stored, and it is held in an index array means 6. The means 7 updates an index value whenever the index is supplied to the means 6. The means 2 transmits a matched length to an encoding data generation means 3, and transmits a difference between the held index and the index held in the entry corresponding to the position of the index  $x(s)$  to an encoding means 4. The means 4 finally converts it into a binary string and updates a record array means 1 when encoding is terminated. A registration control means 5 allocates a number from the means 7 and holds it in the means 6.



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-46235

(43) 公開日 平成9年(1997)2月14日

(51) Int.Cl.<sup>6</sup>

H 0 3 M 7/40

識別記号

庁内整理番号

9382-5K

F I

H 0 3 M 7/40

技術表示箇所

審査請求 有 請求項の数11 O L (全 18 頁)

(21) 出願番号 特願平7-249956

(22) 出願日 平成7年(1995)9月27日

(31) 優先権主張番号 特願平6-239926

(32) 優先日 平6(1994)10月4日

(33) 優先権主張国 日本 (J P)

(31) 優先権主張番号 特願平7-121135

(32) 優先日 平7(1995)5月19日

(33) 優先権主張国 日本 (J P)

(71) 出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72) 発明者 岡村 利彦

東京都港区芝五丁目7番1号 日本電気株式会社社内

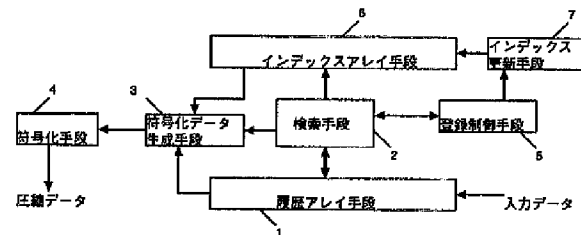
(74) 代理人 弁理士 京本 直樹 (外2名)

(54) 【発明の名称】 データ圧縮装置

(57) 【要約】

【課題】 大きな履歴アレイを用いても、比較すべき文字列の個数、検索手段へ登録する文字列の数が効果的に削減でき、圧縮率の劣化を小さく抑えて符号化速度の向上を図ることができるデータ圧縮方式を提供する。

【解決手段】 入力データを格納する複数のエントリを有する履歴アレイ手段1と、履歴アレイ手段1中の文字列の検索を容易にするための検索手段2と、履歴アレイ手段1の文字列の検索手段2への登録を以下に述べる方法によって制御する登録制御手段と5と、検索手段2に登録されている文字列の先頭位置に対応するエントリにのみインデックスを保持するインデックスアレイ手段6と、インデックスアレイ手段6にインデックスを供給するインデックス更新手段7と、符号化データ生成手段3と、符号化手段4とから構成されている。



## 【特許請求の範囲】

【請求項1】入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、前記履歴アレイ手段内の文字列の前記検索手段への登録を制御する登録制御手段と、を有することを特徴とするデータ圧縮装置。

【請求項2】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする請求項1記載のデータ圧縮装置。

【請求項3】前記登録制御手段は該一致文字列の長さに基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする請求項1または請求項2記載のデータ圧縮装置。

【請求項4】前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の長さが予め決められた閾値以下の場合に該一致文字列の途中の位置から始まる文字列であることを特徴とする請求項3記載のデータ圧縮装置。

【請求項5】前記閾値を動的に変更することを特徴とする請求項4記載のデータ圧縮装置。

【請求項6】前記登録制御手段は該一致文字列の先頭の前記履歴アレイ手段における位置に基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする請求項1または請求項2記載のデータ圧縮装置。

【請求項7】前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の先頭の前記履歴アレイ手段内の位置と入力データ中の該一致文字列の先頭位置との相対距離が予め決められた閾値以上の場合には該一致文字列の途中の位置から始まる文字列であることを特徴とする請求項6記載のデータ圧縮装置。

【請求項8】前記閾値を動的に変更することを特徴とする請求項7記載のデータ圧縮装置。

【請求項9】入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字

列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、入力データ中の該一致文字列の先頭から高々閾値個までの位置から始まる文字列を前記検索手段へ登録する登録制御手段と、を有することを特徴とするデータ圧縮装置。

10 【請求項10】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする請求項9記載のデータ圧縮装置。

20 【請求項11】各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記検索手段に登録されている文字列の先頭のエントリの内、連続する該エントリから成るブロックにインデックスを割り振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に、該エントリが属する該ブロックのインデックスを保持し、前記アレイ手段の位置を表す情報を該エントリが属する該ブロックのインデックスと該エントリの該ブロック内での位置を表す情報から生成される位置コードによって表すことを特徴とする請求項1または請求項9記載のデータ圧縮装置。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は無歪みで復元可能なデータ圧縮復元方式に関する。

【0002】

【従来の技術】レンパルージブ方式として知られるテキスト置換型の無歪みデータ圧縮方式は1980年代後半から広く使用されるようになってきた。中でもLZ77型と呼ばれるデータ圧縮アルゴリズムは圧縮率の良さから計算機上のファイル圧縮ツールにしばしば組み込まれている。LZ77型は既に符号化を終えた入力データを格納するための履歴アレイを持つことを特徴とする。履歴アレイは更新が容易になるように円環的に使用されることが多い。つまり履歴アレイ内に一定長の先読み領域と呼ばれる領域を用意し、そこにはこれから圧縮を行うデータを格納しておき、先読み領域は処理が進むにつれて円環的に移動していく。

【0003】これから圧縮を行うデータ（先読み領域内のデータ）は履歴アレイ中の先読み領域以外の部分から

始まる文字列、つまり既に圧縮が終了している入力データを格納するエントリから始まる文字列と照合し、一致列を求める。十分に長い一致列が見つかったら、入力データの一致列の部分に対する符号語として、履歴アレイ内のその一致列の先頭位置、長さを符号語として送出する。必要に応じてこれらの値は可変長符号化してさらに効率を良くする。一致列の先頭位置はなるべく最新のものに統一し、先読み領域の先頭との相対距離で表すことによって、比較的小さい値に集中させることができ、可変長符号化が有効に働く。履歴アレイ内は常に最新の符号化済みのデータが格納されるように更新していく。つまり、LZ77型は常に最新の符号化済みのデータを“辞書”としており、辞書に登録されている文字列は履歴アレイ内の既に圧縮を終えたデータを格納するエントリから始まる一定長以下の文字列すべてとなる。これはレンパージブ方式のもう一つの形態であるLZ78型の圧縮方法に比べて辞書に登録されている文字列が多いことになり、より長い一致列が見つかり良い圧縮率を示すことが多い。

【0004】十分に長い一致列が見つからなかった場合の処置法としてLZSSと呼ばれるアルゴリズムがある。lha, gzipなどのファイル圧縮用フリーソフトはLZSSを基本としている。LZSSは二つの符号化モードを用いる。十分に長い一致列（1シンボル1byteの場合で3シンボル以上が普通）が見つかった場合に用いるコピーモードと、見つからなかった場合に用いるリテラルモードである。コピーモードの場合は一致長を表すビット列と一致位置を表すビット列が符号語となる。リテラルモードの場合には先頭一文字がそのまま送出される。リテラルモードとコピーモードを区別するために1ビットのフラグをつけるか、または一致長と入力データのアルファベットを合わせて新たに大きなアルファベットを生成するといった処置がとられる。

【0005】LZ77型を実装する上では、これから圧縮を行う入力文字列と履歴アレイ内の文字列との照合処理を高速に行うことが問題となる。図10に示すように、観念的には履歴アレイ手段1に対して何らかの検索手段2を設定し、履歴アレイ内の文字列はすべてこの検索手段2に登録する形になる。なお、図10において、3は符号化データ生成手段、4は符号化手段である。

【0006】現在は専用ハードウェアで並列的に履歴アレイ内を検索する他に、履歴アレイ内が順次更新されることから、二分木、トライなどの木構造を使う検索手段か、ハッシュ表に衝突時用の線形リストを備えた検索手段がよく用いられる。検索手段は文字列の先頭の履歴アレイ内の位置を示すポインタを用いて構成される。検索手段は文字列の先頭位置を示すポインタから履歴アレイ内の文字列を読み込みターゲットの文字列と比較する。

【0007】二分木を用いた方法はIEEEの刊行物「IEEE トランザクション オンコミュニケーション

」1986年vol. 34, no. 12, pp. 1176-1182に開示されている。トライを用いた方法は米国特許第4,906,991号明細書に詳しく述べられている。ハッシュ表+線形リストの方法は特開平3-68219号公報に詳しく述べられている。

【0008】それぞれの検索法を図11～図14に示す。図11は履歴アレイ手段の状態を表し、図12、図13、図14は履歴アレイ手段が図11の状態のときの各検索手段の構成を表す。図12は二分木を用いた方法、図13はトライを用いた方法、図14はハッシュテーブルに衝突用の線形リストを備えた方法である。アレイから溢れた文字列は検索手段から削除する必要があるため、検索手法は削除のコストが低いものでなければならない。

【0009】

【発明が解決しようとする課題】LZ77型では、長い一致列が見つければ圧縮率は向上する。よりたくさんの文字列とマッチングをとればより長い一致列が見つかる可能性が高くなり、そのためには履歴アレイの大きさN、コピーの最大長Lを大きくすればよい。しかし、N、Lを大きくすると三つの問題が生じる。

【0010】第一にN、Lの増大は文字列の位置、長さを表すために必要なビット数の増加につながり、かえって圧縮率を劣化させる恐れがある。しかし、この問題は位置、長さなどを可変長符号化することである程度解決できる。1シンボルを1バイト（8ビット）とした時、位置、長さを固定長符号化するという前提の下では履歴アレイ長N=8192程度、一致列の最大長L=64程度が最良と言われていたが、現在ではN=32768、65536、L=256～1024程度が最もよい圧縮率を示すことが判明している。

【0011】第二にN、Lの増大は符号化時にメモリを大きく消費することにつながる。しかし、N=65536、L=2048程度であつたら検索手段まで含めて高々2メガバイトのメモリで済むため、現在ではワークステーションや高性能なパソコンで実行する上では問題なく実行できる。

【0012】第三の問題が最も重大で、N、Lの増大は符号化速度を著しく劣化させる。LZ77型の符号化速度を見積もる指標として文字比較回数がある。データ長をMとすると、二分木、トライ、ハッシュテーブル+線形リストの文字比較回数は次のように近似される。

【0013】

- ・特別に検索手段を設けない … M'LN
- ・二分木 … MLlog<sub>2</sub>N
- ・トライ … ML
- ・ハッシュ表 … M'LN'

【0014】LZ77型では入力データは部分列に分解され、部分列毎に符号語が対応することになるが、M'はこの部分列の個数である。またN'は線形リストの平

均の長さである。二分木やトライを用いた場合、文字列を検索手段に登録するためには検索手段を用いて検索を行い、しかるべき位置に挿入しなければならない。そのために、データ中の全ての位置から一定長以下の文字列を検索手段を用いて検索しなければならない。これに対してハッシュテーブルを使用する方法では文字列検索を必要とするのは、符号化を行う文字列の先頭のみからに限られる。一致列の途中から始まる文字列の検索手段への登録に関してはハッシュ関数の計算とハッシュ表の1レコードの書き換えだけで済み、文字列検索を必要とし

10 ない。そこでハッシュテーブルを用いた方法では文字列比較回数はMではなく、M'の倍数という形で表せるのである。N'は最悪Nに等しくなるが、十分なハッシュテーブルを用意すれば平均は十分に小さく、また、圧縮率の多少の劣化を許容すればN'は適当な回数で打ち切ることができるので小さい定数で抑えることもできる。そのためにM'と二分木、トライの場合のMの差が効いて、ハッシュテーブルを使用した方法の高速性が生まれる。とくに冗長の大きなデータの場合にはMとM'の差が大きく、ハッシュテーブルを使用した場合と木構造を用いた場合とでは実行速度の大きな違いが生じる。N=65536, L=1024に設定すると二分木、トライでは著しく符号化速度が劣化してしまい、ハッシュテーブルを使用した方法の独壇場となる。しかし、ハッシュテーブルを使用した方法で最長一致列を求めようとするとN'が大きくなり、1回の検索にかかるコストが余りにも大きく、特別に検索手段を設けない場合と変わらなくなり、実行速度は極めて劣化してしまう。

【0015】検索手段に木構造を用いた場合の高速化手法の一つが米国特許4,906,991号の拡張具体例に開示されている。そこで述べられている方法は、検索手段にトライ構造を用い、符号化する文字列の先頭から始まる文字列のみを検索手段に登録する方法である。また、一致長が2または3の場合にその途中から始まる文字列を検索手段に登録する方法についても簡単に述べられている。

【0016】本発明の目的は、大きな履歴アレイを用いても、比較すべき文字列の個数、検索手段へ登録する文字列の数が効果的に削減でき、圧縮率の劣化を小さく抑えて符号化速度の向上を図ることができるデータ圧縮方式を提供することにある。本発明は先に述べた米国特許4,906,991の方法を拡張した、より柔軟性のある登録制御を行うことによって目的を達成する。

【0017】

【課題を解決するための手段】第1の発明は、入力データを格納する複数のエントリを有する履歴アレイ手段を有し、これから圧縮を行う入力データと、前記履歴アレイ手段に格納されている文字列を比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で

表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、前記履歴アレイ手段内の文字列の前記検索手段への登録を制御する登録制御手段と、を有することを特徴とする。

【0018】第2の発明は、第1の発明のデータ圧縮装置において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする。

【0019】第3の発明は、第1または第2の発明において、前記登録制御手段は該一致文字列の長さに基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする。

【0020】第4の発明は、第3の発明において、前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の長さが予め決められた閾値以下の場合に該一致文字列の途中の位置から始まる文字列であることを特徴とする。

【0021】第5の発明は、第4の発明において、該閾値を動的に変更することを特徴とする。

【0022】第6の発明は、第1または第2の発明において前記登録制御手段は該一致文字列の先頭の前記履歴アレイ手段における位置に基づいて前記検索手段に登録される新たな文字列を決定することを特徴とする。

【0023】第7の発明は、第6の発明において前記検索手段に新たに登録される文字列は入力データ中の該一致文字列の先頭から始まる文字列と、該一致文字列の先頭の前記履歴アレイ手段内の位置と入力データ中の該一致文字列の先頭位置との相対距離が予め決められた閾値以上の場合には該一致文字列の途中の位置から始まる文字列であることを特徴とする。

【0024】第8の発明は、第7の発明において、該閾値を動的に変更することを特徴とする。

【0025】第9の発明は、入力データを格納する複数のエントリを有する履歴アレイ手段を持ち、これから圧縮を行う入力データと前記履歴アレイ手段に格納されている既に符号化を終えている入力データを比較し、十分に長い一致文字列が発見された場合には該一致文字列を前記履歴アレイ手段における該一致文字列の位置、長さの情報で表すことによって圧縮を達成するデータ圧縮装置において、前記履歴アレイ手段の文字列の検索を行うための検索手段と、入力データ中の該一致文字列の先頭から高々閾値個までの位置から始まる文字列を前記検索手段へ登録する登録制御手段と、を有することを特徴とする。

【0026】第10の発明は、第9の発明のデータ圧縮方法において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記履歴アレイ手段のエントリの内、前記検索手段に登録される文字列の先頭のエントリにのみインデックスを振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に該インデックスを保持し、前記アレイ手段中の位置を表す情報を該インデックスから生成される位置コードによって表すことを特徴とする。

【0027】第11の発明は、第1または第9の発明において、各エントリが前記履歴アレイ手段の各エントリに対応するインデックスアレイ手段を持ち、前記検索手段に登録されている文字列の先頭のエントリの内、連続する該エントリから成るブロックにインデックスを割り振り、前記インデックスアレイ手段は前記履歴アレイ手段の該エントリに対応する位置に、該エントリが属する該ブロックのインデックスを保持し、前記アレイ手段の位置を表す情報を該エントリが属する該ブロックのインデックスと該エントリの該ブロック内での位置を表す情報から生成される位置コードによって表すことを特徴とする。

#### 【0028】

【作用】本発明では検索手段への履歴アレイ内の文字列の登録を制御するという形で、比較する文字列の個数と検索手段へ登録する文字列の個数を削減させている。圧縮率を良くするための大きな履歴アレイでは同じ文字列が頻繁に現れることが多く、符号化速度の面から大きな負荷を与えるが、本発明はこれを解消するための手段を与える。

【0029】圧縮を行う文字列は履歴アレイ内の文字列とのマッチングをとって一致列を求める。このように符号化する文字列の場合、検索は必須で、その結果を利用して検索手段に容易に挿入することができるので、符号化の先頭位置から始まる文字列については検索手段に登録するのが妥当である。一致列が見つかった場合、符号化された文字列の途中の位置から始まる文字列を処理する方法が本発明の着眼点である。登録制御手段の判断の基準は、復元側でもわかる仕組みでなければならない。そこで、一致長や一致列の位置といった情報から生成される基準を使用する方法が考えられる。

【0030】一致長が長い文字列が符号化された場合、その途中から始まる文字列に関しても既に履歴アレイ内に存在しているものが多いことになる。そこで途中から始まる文字列の登録を抑えても圧縮率に大きく影響しないことが期待できる。本発明の第3から第5の発明はこのことを利用した方式である。同じような文字列が繰り返し現れることが冗長度の高いデータに対しては、検索構造への無駄な登録を大きく省くことができ、履歴アレイ内から始まる文字列をすべて登録していた従来の方式

に比べて符号化速度を向上させることができる。

【0031】一致文字列の履歴アレイ内での位置に基づいて符号化された文字列の途中の位置から始まる文字列を登録するかを判断するという手法は次のようになる。一致列が昔に現れたものであったら、その列は近い内に履歴アレイから溢れてしまう。そのような状況になってもその文字列が絡んだ文字列が参照できるようにするために、途中から始まる文字列を登録するのである。逆に比較的近い位置に一致列が同定された場合、その符号化文字列が絡んだ文字列はしばらくは履歴アレイ内に存在するので、あえて符号化された文字列の途中から始まる文字列まで検索手段に登録しなくても圧縮率はそれ程劣化しないことが期待できる。冗長なデータでは比較的近い位置に最長一致列が見つかることが多いため、検索手段に登録される文字列を絞ることができ、従来の方式に比べて符号化速度を向上させることができる。

【0032】また、単純に一致文字列の先頭から閾値個までの位置から始まる文字列を登録する方法が考えられる。ただし、閾値が一致文字列より大きい場合は一致文字列のすべての位置から始まる文字列を登録する。そういった意味ではこの方法も、一致長に基づいて登録される文字列が決めている、とみなすこともできる。この場合、閾値を小さく設定すると登録される文字列が少なくなり、符号化速度の向上を生む。

【0033】本発明の方式では、履歴アレイ中のすべての位置から始まる文字列を参照することができるわけではないので、一致列の先頭位置をアレイ中の位置でそのまま表すと冗長になる。必要な位置にだけインデックスを振り、一致列の先頭位置のインデックスと最新のインデックスの差を用いて位置を表すことによって、履歴アレイの位置で見た相対距離で表すより小さな値で表現でき、位置を表す符号語のビット長に関しては有利になる。第2の発明のように、インデックスアレイを用いて、各エントリに対応するインデックスを保持する必要がある。また、検索手段に登録されている文字列の先頭位置の中で連続するものをまとめたブロックを形成し、ブロックに割り振ったインデックスを用いて一致列の先頭位置を表すこともできる。

#### 【0034】

【発明の実施の形態】次に、本発明の実施例について図面を参照して説明する。

【0035】図1は、一実施例のデータ圧縮装置を示す構成図である。この装置は、入力データを格納する複数のエントリを有する履歴アレイ手段1と、履歴アレイ手段1中の文字列の検索を容易にするための検索手段2と、履歴アレイ手段1の文字列の検索手段2への登録を以下に述べる方法によって制御する登録制御手段5と、検索手段2に登録されている文字列の先頭位置に対応するエントリにのみインデックスを保持するインデックスアレイ手段6と、インデックスアレイ手段6にインデッ

クスを供給するインデックス更新手段7と、符号化データ生成手段3と、符号化手段4とから構成されている。

【0036】次に、このデータ圧縮方式の動作を図2のフローチャートを参照しながら説明する。

【0037】入力データを $X = x(0) x(1) x(2) \dots$ とする。履歴アレイ手段1はNシンボルの入力データを格納するとする。1シンボル1バイト(8ビット)として、 $N = 32768$ または $65536$ が適当である。Nは任意の値でよいが円環的に用いるため、2の中ですることがよく行われる。また、一致長の最大長Lは256以上にとる。履歴アレイ手段における先読み領域はL以上の大きさにする。この実施例では先読み領域の大きさをLとする。

【0038】まず、履歴アレイ手段1、検索手段2、インデックスアレイ手段6、インデックス更新手段7を初期化する(ステップS1)。

【0039】履歴アレイ手段の初期状態は空でも適当なデフォルトの文字列でも、要は復元時に同じ状態が再現できるものであったら何でもよい。検索手段2、インデックスアレイ手段6、インデックス更新手段7の初期状態はアレイ手段の初期状態に合わせて決まる。

【0040】今、 $\dots x(s-2) x(s-1)$ までの符号化を終えて $x(s) x(s+1) \dots$ という文字列の符号化を行うものとする(ステップS2)。このとき履歴アレイ手段内の各エントリには $x(s-N+L), \dots, x(s-1), x(s), \dots, x(s+L-1)$ という文字列が格納されており、特に先読み領域には $x(s), \dots, x(s+L-1)$ という文字列が格納されている。

検索手段2に登録されている文字列は $x(s-N+1), \dots, x(s-1)$ の内、限定された位置から始まる、長さL以下の文字列である。検索手段2に登録されている文字列の先頭位置に対応するインデックスアレイ手段6のエントリには対応するインデックスが保持されている。

【0041】検索手段2を用いて文字列 $x(s) x(s+1) \dots$ と検索手段2に登録されている文字列と比較し(ステップS3)、最長一致列を求める。場合によっては最長一致列ではなくなるべく長い一致列でよい。 $x(s) x(s+1) \dots$ は検索結果が分かっているので検索構造へ登録する(ステップS4)。

$x(s)$ が格納されているエントリにはインデックス更新手段7から供給されるインデックスを割り振り、インデックスアレイ手段6に保持する。インデックス更新手段7はインデックスアレイ手段6にインデックスを供給する度にインデックス値を1更新する。インデックスは各エントリを区別できればよいので、Nに対する剰余で表すことにより0からN-1で表すことができる。

【0042】最長一致列が $x(s-N+L+k) \dots x(s-N+L+k+1-1) = x(s) \dots x(s+1-1)$ であったとする(ステップS5)。検索手段2は、

一致長Lを符号化データ生成手段3に送る。Lが予め決められた一定値M以上であったら、符号化データ生成手段3は一致列が見つかったというフラグとともに $x(s) \dots x(s+1-1)$ に対応する符号語としてLと $x(s-N+L+k)$ の位置に対応するエントリに保持されているインデックスと $x(s)$ の位置に対応するエントリに保持されているインデックスとの差(位置コード)を符号化手段4に送出する(ステップS6、これをコピーモードによる符号化と呼ぶ)。

LがMよりも小さかったら $L=1$ として、符号化データ生成手段は一致列が見つからなかったというフラグとともに $x(s)$ をそのまま符号化手段に送出する(ステップS7、これをリテラルモードによる符号化と呼ぶ)。

【0043】符号化手段4はL, d,  $x(s)$ などを最終的な二進列に変換する(ステップS6, S7)。符号化手段4には固定長のものの他、ハフマン符号化、算術符号化などを利用できる。符号化手段4はデータが入力される度に符号化を行う他、バッファに値を一時的に蓄え、ブロック単位に処理することもできる。

【0044】 $x(s) \dots x(s+1-1)$ の符号化が終わった時点で、履歴アレイ手段1を更新する。新たな入力データを読み込み、先読み領域は $x(t+1)$ のエントリから始まる領域に移動する。このとき、履歴アレイ手段1から削除された古い文字列は検索手段2からも削除される(ステップS8)。

【0045】登録制御手段5は $x(s+1), \dots, x(s+1-1)$ から始まる文字列のうち、検索手段2に登録するものがあるかどうかを判断する。例えばリテラルモードで符号化された文字列の場合は登録するものがないと判断して、次の文字列の符号化に移る(ステップS9)。登録すると選択された文字列は検索手段2へ登録し、それぞれの先頭位置のエントリに順にインデックス更新手段7から供給される番号を振り、インデックスアレイ手段6に保持する(ステップS10)。

【0046】以上で入力データの部分列 $x(s) \dots x(s+1-1)$ の符号化は終了で、履歴アレイ手段1に新しいシンボルを挿入し、 $x(s+1)$ から始まる文字列の符号化を開始する。

【0047】登録制御において一致長を基準にする場合、一致長が閾値よりも大きかったら登録を控えるのが妥当である。閾値は経験的には4から16程度が良い結果をもたらす。動的に閾値を変化させる場合は、一致長のそれまでの平均値を閾値とする方法が考えられる。図3は平均値を用いて閾値を動的に変化させる場合の登録制御手段の処理の流れを示すフローチャートである。

【0048】図4は従来方式と一致長を基準にする登録制御方式を比較した図である。図2の処理の流れに沿って、図4の状況での一致長を基準とした登録制御の具体例を説明する。履歴アレイ手段22は12個のエントリから成っている。履歴アレイ手段22の中で灰色の部分

は先読み領域で、これから符号化を行う文字列を格納しており、今は履歴アレイ手段の8から11の位置のエントリである。履歴アレイ手段22の先読み領域以外の位置から始まる文字列はすべて検索手段に登録されているとして、インデックスアレイ手段21は‘0’から順に‘7’までのインデックスを格納しているとする。先読み領域の長さが4のため、文字列の検索手段への登録は、ある位置から始まる長さ4の文字列を登録する、という形で行う。このとき、履歴アレイ手段22内で先読み領域の先頭から始まる文字列（圧縮を行おうとしている文字列）と検索手段に登録されている文字列との間で最長一致列を求めると、“b c b”という文字列が得られる（ステップS3）。このとき8から始まる文字列は、一致列を求める過程で検索手段に登録される。インデックスアレイ手段21の対応するエントリには新しいインデックスである‘8’が格納される（ステップS4）。“b c b”に対する符号語は一致長‘3’（先読み領域の先頭のインデックスと比較して3個前のインデックスで示される位置、という情報）を元に生成される（ステップS6）。符号語の出力が済んだら、履歴アレイ手段22を3文字分更新する。先読み領域は11から始まる領域に移動する。履歴アレイ手段を円環的に使用しているため、先読み領域は（11, 0, 1, 2）の位置のエントリに移動する。0, 1, 2から始まる文字列は検索手段から削除される。その位置のインデックスも無効になる。履歴アレイ手段22の（0, 1, 2）の位置には新たに読み込んだデータである“c a a”が挿入される（ステップS9）。図4（a）は登録制御が行われない場合、図4（b）は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、24または26で表される。登録制御では、履歴アレイ手段24（または26）の9, 10から始まる文字列（“c b b c”, “b b c a”）を検索手段に登録するかどうか問題になる（ステップS9）。従来方式は9, 10から始まる文字列を検索手段に登録する（図4（a））。対応するインデックス‘9’, ‘10’がインデックスアレイ手段23に格納される。一致長を基準にした登録制御では、一致長が3のため、閾値が4以上であつたら従来方式と同様に検索手段への登録、インデックスアレイ手段23の更新を行い（図4（a）、ステップS10）、閾値が4よりも小であつたら9, 10から始まる文字列の登録は行わない（図4（b））。インデックスアレイ手段25の9, 10のエントリは空になる。図4（a）の状況で、次に11の位置に格納されるインデックスは‘11’であるが、図4（b）では‘9’となる。

【0049】登録制御において一致列の開始点を基準にする場合、一致列の開始点の位置が閾値よりも小さかつたら登録を控えるのが妥当である。動的に閾値を変化させる方法は一致長の場合と同様に平均値を使用する方法

が考えられる。

【0050】図5は従来方式と一致位置を基準にする登録制御方式を比較した図である。図5の状況での一致位置を基準とした登録制御の具体例を説明する。図5の履歴アレイ手段32やインデックスアレイ手段31の状況は図4と同様である。また、最長一致列を求め、履歴アレイ手段を更新する状況も図4と同様である。図5

（a）は登録制御が行われない場合、図5（b）は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、34または36で表される。登録制御では、履歴アレイ手段34（または36）の9, 10から始まる文字列（“c b b c”, “b b c a”）を検索手段に登録するかどうか問題となる。従来方式は図4の場合と同様に履歴アレイ手段34の9, 10から始まる文字列を検索手段に登録する（図5（a））。対応するインデックス‘9’, ‘10’がインデックスアレイ手段33に格納される。一致位置を基準にした登録制御では、一致位置が先読み領域の先頭からみて3個前の位置のため、閾値が4以上であつたら従来方式と同様に検索手段への登録、インデックスアレイ手段33の更新を行い（図5（a））、閾値が4よりも小であつたら9, 10から始まる文字列の登録は行わない（図5（b））。インデックスアレイ手段35の9, 10のエントリは空になる。図5（a）の状況で、次に11の位置に格納されるインデックスは‘11’であるが、図5（b）では‘9’となる。

【0051】次に、本発明の登録制御手段のもう一つの実施例について説明する。図6はこの実施例の登録制御の流れを示す図である。先頭x（s）から始まる文字列は先に述べたようにこの符号化のための検索結果を用いて既に登録されているとする。今一致列がx（s）…x（t）（一致長 $l = t - s + 1$ ）であつたとする（ステップT1）。リテラルモードで符号化された場合（ $l = 1$ ）、または閾値 $r = 1$ の場合には先頭のx（s）から始まる文字列のみが検索手段2に登録されるので、一致列の途中から始まる文字列の登録はない（ステップT2）。そうでない場合、一致列として切り出された文字列の中で、最初のr個の位置から始まる文字列のみを検索手段2に登録する。つまり、x（s）…x（t）（ $l = t - s + 1$ ）の途中から始まる文字列としては、x（s+1）, …, x（s+r-1）から始まる文字列のみを検索手段2に登録するよう登録制御を行うのである（ステップT4）。ただし、 $l < r$ の場合にはx（s+1）, …, x（s+l-1）（=x（t））から始まる文字列を検索手段2に登録する（ステップT3, T5）。閾値rの設定を変更することで符号化速度と圧縮率のどちらを重視するかを調節するようなアプリケーションも考えられる。この場合、圧縮データの先頭にこの閾値を記しておくことにより、復元側ではこの閾値を知ることができ、この値さえ知っていれば、履歴アレイ中のどの



位置から始まる文字列が登録されているのかを正しく知ることができるため正しく復元できる。

【0052】図7は従来方式と一致列の先頭から閾値個までの文字列を登録する方式を比較した図である。図7の状況での本登録制御方式の具体例を説明する。図7の履歴アレイ手段42やインデックスアレイ手段41などの状況は図4と同様である。また、最長一致列を求め、履歴アレイ手段を更新する状況も図4と同様である。図7(a)は登録制御が行われない場合、図7(b)は登録制御が行われる場合の状況を表している。両者とも履歴アレイ手段の状態は同一で、44(または46)で表される。登録制御では、履歴アレイ手段44(または46)の9, 10から始まる文字列(“c b b c”, “b b c a”)を検索手段に登録するかどうか問題になる。従来方式は履歴アレイ手段44の9, 10から始まる文字列を検索手段に登録する(図7(a))。対応するインデックス‘9’, ‘10’がインデックスアレイ手段43に格納される。閾値を2とした場合、先頭の次の位置である、履歴アレイ手段46の9の位置から始まる文字列が検索手段に登録される(図7(b))。インデックスアレイ手段46の9の位置のエントリは‘9’が格納され、10の位置のエントリは空となる。図7(a)の状況で、次に11の位置に格納されるインデックスは‘11’であるが、図7(b)では‘10’となる。

【0053】また、更に閾値 $r$ が2以上のときには次のように圧縮率の改善をみることができる。 $r \geq 2$ の時には符号化する文字列の先頭 $x(s)$ の次の文字 $x(s+1)$ から始まる文字列を検索手段2に登録する必要があるため、木構造などを検索手段に用いた場合にはその過程で $x(s+1)$ から始まる文字列に対する一致列を求める必要がある。そこで、 $x(s)$ からの一致列と $x(s+1)$ からの一致列を比較して $x(s+1)$ からの一致列の方が大きかったら $x(s)$ をリテラルモードで符号化し、 $x(s+1)$ からの一致列をコピーモードで符号化する。こうすることによって、2回のコピーモードで符号化するところを1回のリテラルモードと1回のコピーモードで符号化できる場合が多い。リテラルモードの方がコピーモードよりも少ないビット数で表現できるため圧縮率が向上するのである。この処理の詳細な説明を図8に示す。

【0054】 $x(s)$ からの一致列が $x(s) \cdots z(s+1-1)$ ( $l$ は一致長)であったとする(ステップU1)。 $l$ が十分に大きくないときには $x(s)$ をリテラルモードで符号化し(ステップU3)。履歴アレイ手段1の更新、履歴アレイ手段1から溢れた文字列を検索手段2から溢れた文字列の削除を行い(ステップU8)、 $x(s+1)$ の符号化に移る(ステップU13)。 $l$ が十分に大きいときには、先頭の次の文字 $x(s+1)$ から始まる文字列と履歴アレイ手段1内の文字列との最長

一致列を求め、その検索結果を用いて $x(s+1)$ から始まる文字列を検索手段2に登録する(ステップU4)。一致列を $x(s+1) \cdots x(s+1')$ ( $l'$ は一致長とする)。 $l$ と $l'$ を比較し(ステップU5)、 $l'$ の方が大きかったら $x(s)$ をリテラルモードで、 $x(s+1) \cdots x(s+1')$ をコピーモードで符号化する(ステップU9)。符号化された分の大きさ( $(1+1')$ 個の文字分)履歴アレイ手段1を更新し、履歴アレイ手段1から溢れた文字列の削除を行う(ステップU9)。 $x(s+1) \cdots x(s+1')$ から始まる文字列に対して図6に示した登録制御法で検索手段2への登録を行う(ステップU11)。そして、 $x(s+1'+1)$ から始まる文字列の符号化に移る(ステップU14)。 $l$ が $l'$ 以上の場合には $x(s) \cdots x(s+1-1)$ をコピーモードで符号化する(ステップU7)。符号化された分の大きさ(1個の文字分)履歴アレイ手段を更新し、履歴アレイ手段1から溢れた文字列の削除を行う(ステップU10)。 $x(s) \cdots x(s+1-1)$ から始まる文字列に対して図6に示した登録制御法で検索手段2への登録を行う(ステップU12)。このとき、 $x(s+1)$ から始まる文字列の登録は既に済んでいることに注意。そして、 $x(s+1)$ から始まる文字列の符号化に移る(ステップU15)。

【0055】以上のように登録制御を行うと、検索手段に登録されている履歴アレイ手段内の文字列は連続して現れることが多くなる。ここで“連続して現れる文字列”は、隣合う位置から始まる文字列、ということを意味する。次に、このことを利用して、圧縮率を向上させる方式を図9を用いて説明する。

【0056】今、履歴アレイ手段内で $x(p)$ ,  $x(p+1)$ ,  $\cdots$ ,  $x(r)$ から始まる文字列は検索手段に登録されており、 $x(p-1)$ ,  $x(r+1)$ から始まる文字列は検索手段に登録されていないとする。このとき、 $x(p) \cdots x(r)$ を“ブロック”と呼ぶことにする。図9に示すように、履歴アレイ手段52の先読み領域以外のエントリは、いくつかのブロックと検索手段に登録されていない部分に分けられることになる。このとき現れたブロックには順にインデックスを割り振り、インデックスアレイ手段51はこのブロックのインデックスを格納する。図9に示したように、同じブロック内のエントリは同一のインデックスをインデックスアレイ手段51に格納している。従来方式では履歴アレイ手段内の先読み領域以外の位置から始まるすべての文字列が検索手段に格納されているため、先読み領域以外の部分は一つの大きなブロックになる。

【0057】先読み領域内の文字列を $x(s) x(s+1) \cdots$ とし、 $x(s) x(s+1) \cdots$ と履歴アレイ手段内で検索手段に登録されている文字列との最長一致列が $x(s-k) \cdots x(s-k+1-1)$ ( $=x(s) \cdots x(s+1-1)$ )であったとする。前述の方式では一致

位置の情報を  $x(s-k)$  に対応するインデックスに基づいて生成したが、このブロックを用いた方式では、 $x(s-k)$  の属するブロックのインデックスと  $x(s-k)$  のそのブロック内での位置の情報に基づいて生成する。具体的には、 $x(s-k)$  が先読み領域からみて  $r$  個前のブロックに属し、そのブロックの最後のエントリには  $x(s-k+u)$  が格納されているとすると、

$(r, u)$  という情報で  $x(s-k)$  の位置を指定する。図9は一致長が5で、 $r=3$ 、 $u=0$ である状況を表している。 $r$  はインデックスアレイ手段51から求められる。 $u$  はブロックのインデックスに対してそのブロックの最後のエントリの位置を格納するアレイを用意する方法か、またはインデックスアレイ手段51を用いて同じインデックスが続くところまで検索する方法によって求めることができる。

【0058】一般に複数のエントリが集まってブロックを形成するため、ブロックの数は検索手段に登録されているエントリの数に比べてはるかに小さいものとなる。ブロックのインデックス情報（一致列は  $r$  個前のブロックから始まる、という  $r$  の値）の分布は、個々のエントリに割り振ったインデックス情報（一致列は  $r'$  個前のインデックスの位置から始まる、という  $r'$  の値）の分布に比べて小さい値に集中するため、ブロックのインデックスの符号長の方が小さい値で表せることになる。一方、ブロックのインデックスを用いる方法はブロック内の位置を表す情報も付加しなければならないため、全体としてみれば個々のエントリにインデックスを割り振る方法と圧縮率は変わらない場合もある。しかし、例えば次に示すような場合に圧縮率の改善をみることができる。

【0059】データ中にランダムに近い大きな領域があると、リテラルモードの出力が続く。つまりその部分の文字列は連続して検索手段に登録され、大きなブロックを形成する。しかし、そのブロックから始まる文字列はその後参照される可能性は低い。各エントリにインデックスを割り振った場合には、そのような大きなブロックより遠い位置に割り振られたインデックスは、最新のインデックスと大きく離れた値となる。しかし、ブロックのインデックスは最新のブロックのインデックスとそれほど離れた値とはならない。ブロックのインデックスを用いて一致位置を表す方法は、このように大きな無駄な領域を飛び越えて一致位置を効果的に表す方法となるのである。

【0060】検索手段として二分木を用い、履歴アレイ手段の大きさ  $N=16384$ 、一致列の最大長  $L=256$  として2メガバイトのCプログラムソースを圧縮する実験を行った。従来の方法では175秒の実行時間で28.1%の圧縮率を達成した。図3に示した登録制御を行った本発明では、32秒の実行時間で29.2%の圧縮率を達成した。つまり、1%未満の圧縮率の劣化で6

倍の符号化速度の向上を得ている。また、図8に示した本発明において閾値を2に設定して登録制御処理を行った本発明の方式では32秒の実行時間で28.7%の圧縮率を達成し、図3の登録制御法と同じ圧縮時間で0.5%程度優れた圧縮率を達成している。また、 $N=8192$ 、 $L=32$ と変更して従来の方法と同じデータを圧縮したときには、67秒の実行時間で31.0%の圧縮率であった。つまり、検索手段として二分木を用い、高速な符号化を求めるのなら、 $N$ 、 $L$ を変更するよりも、本発明の方式を採用した方が有効であり、かつ圧縮率の劣化も小さいことがわかる。

【0061】また、図9に示したブロックのインデックスを用いた符号化法で等長符号化を行った場合の効果は次のように評価できる。検索手段に登録されているエントリの数を  $M$ 、ブロックの平均長を  $B$  とする。ブロックの数は  $M/B$  となる。個々のエントリにインデックスを順に割り振った場合、インデックスを表すためのビット数はほぼ  $\log M$  ビットとなる。一方、ブロックのインデックスを表すためのビット数はほぼ  $\log(M/B) = \log M - \log B$  ビットとなる。ここで  $\log$  の底は2である。ブロックがどれも同じ大きさであったら、ブロック内での位置を表すためにほぼ  $\log B$  ビット必要になり、この場合はブロックのインデックスを用いた方式を用いても圧縮率は変わらない。しかし、履歴アレイ手段中にランダムに近い大きな領域がある場合、その領域からなるブロックを除いた、実際に後に参照される文字列を含むブロックの平均長  $B'$  は  $B$  よりも小さくなる。そこで、ブロックのインデックスを用いる方式によって一回の一致位置情報の出力につき  $(\log B - \log B')$  ビットだけ圧縮データを短縮化できるのである。

【0062】以上本発明のデータ圧縮方式について説明したが、本発明は、データ復元方式にも応用できる。データ復元方式の実施態様を以下に説明する。

(1) 既に復元を終えたデータを格納する複数のエントリを有する履歴アレイ手段を有し、前記履歴アレイ手段における位置、長さを表す圧縮符号語に対応する文字列を複製することで元データを復元するデータ復元方式において、各エントリが前記履歴アレイ手段の各エントリに対応したインデックスアレイ手段を有し、圧縮データに対応する可能性がある文字列は前記履歴アレイ手段の制限された位置から始まる文字列のみであり、前記インデックスアレイ手段は、圧縮データに対応する可能性がある文字列の先頭位置にのみ割り振られたインデックスを保持し、圧縮データにおける前記インデックスから生成された圧縮符号語から複製される文字列の開始点を確定することを特徴とするデータ復元方式。

(2) 圧縮データに対応する可能性がある前記履歴アレイ手段内の文字列の先頭位置は、前記復元文字列の長さに基づいて決定されることを特徴とする(1)記載のデ

ータ復元方式。

(3) 圧縮データに対応する可能性がある前記履歴アレ  
イ手段内の先頭位置は、復元された文字列の先頭位置と  
前記復元文字列の長さが予め決められた閾値以下であ  
った場合の前記復元文字列の途中の位置であることを特徴  
とする(2)記載のデータ復元方式。

(4) 前記閾値を動的に変更することを特徴とする

(3)記載のデータ復元方式。

(5) 圧縮データに対応する可能性がある文字列の先頭  
位置は、圧縮データで指示される前記履歴アレイ手段内  
の復元文字列と同一の文字列の先頭の前記アレイ手段に  
基づいて決定されることを特徴とする(1)記載のデー  
タ復元方式。

(6) 圧縮データに対応する可能性がある前記履歴アレ  
イ手段内の文字列の先頭位置は、復元文字列の先頭位置  
と、圧縮データによって指示される復元文字列と同一の  
文字列の先頭の前記履歴アレイ手段における位置と、前  
記復元文字列の先頭の位置との相対距離が予め決められ  
た閾値以上であった場合の前記復元文字列の途中の位置  
であることを特徴とする(5)記載のデータ復元方式。

(7) 閾値を動的に変更することを特徴とする(6)記  
載のデータ復元方式。

(8) 圧縮データに対応する可能性がある前記履歴アレ  
イ手段内の先頭位置は、過去に復元された文字列の中で  
先頭位置から閾値以下の順位の位置に制限されているこ  
とを特徴とする(1)記載のデータ復元方式。

(9) 前記履歴アレイ手段中の圧縮データに対応する可  
能性がある文字列うち、連続している文字列の先頭エン  
トリをブロックとしてまとめ、同じ該ブロックに属する  
エントリには同じインデックスを割り振り、前記インデ  
ックスアレイ手段はこれを格納し、圧縮符号語から該ブ  
ロックのインデックスと該ブロック内での位置の情報が  
求められ、一致列の開始位置が定まることを特徴とする

(1)記載のデータ復元方式。

#### 【0063】

【発明の効果】以上説明したように本発明のデータ圧縮  
方式を用いれば、大きな履歴アレイを用いても比較すべ  
き文字列の個数、検索手段へ登録する文字列の数が削減  
でき、符号化速度の向上を図ることができる。特に冗長  
の大きなデータに対しては大きな符号化速度の向上を生  
む。更に、登録制御を行うことを積極的に利用して、検  
索手段に登録されている文字列の先頭エントリ、または  
隣接する先頭エントリをまとめたブロックに割り振られ  
たインデックスを用いて一致位置情報を表すことによ  
り、圧縮率の劣化を小さく抑えることができる。このよ

うに本発明によって圧縮率を劣化させずに符号化速度の  
向上を達成できるのである。

【0064】また、登録に検索と同じ時間が必要となる  
木構造を用いた検索手段に関しては本発明のように選択  
的に検索構造へ登録することは極めて効果がある。ハッ  
シュテーブルを用いた方法でも、ハッシュ関数を計算す  
る回数が少なくなるとともに、検索すべきアレイ内の文  
字列の数も減少し、線形リストを辿る回数が減少するた  
めに効果がある。このように、本発明の登録制御は様々  
な検索手段に対応させることができる。

#### 【図面の簡単な説明】

【図1】本発明の一実施例を示すブロック図である。

【図2】本発明の処理の流れを示す図である。

【図3】動的に閾値を変更する処理の流れを示す図であ  
る。

【図4】一致長を基準にして登録制御を行う方式の具体  
例を、従来方式と比較して示した図である。

【図5】一致位置を基準にして登録制御を行う方式の具  
体例を、従来方式と比較して示した図である。

【図6】閾値個以下の位置から始まる文字列のみを検索  
手段に登録する場合の登録制御手段の処理の流れを示す  
図である。

【図7】閾値個以下の位置から始まる文字列のみを検索  
手段に登録する方式の具体例を、従来方式と比較して示  
した図である。

【図8】閾値個以下の位置から始まる文字列のみを検索  
手段に登録する方式で、閾値が2以上の場合の改良型の  
登録制御手段および符号化データ生成手段の処理の流れ  
を示す図である。

【図9】ブロックのインデックスによって一致位置を表  
す方式を示す図である。

【図10】従来の圧縮方式の構成を示す図である。

【図11】履歴アレイ手段を示す図である。

【図12】二分木を用いた方法を示す図である。

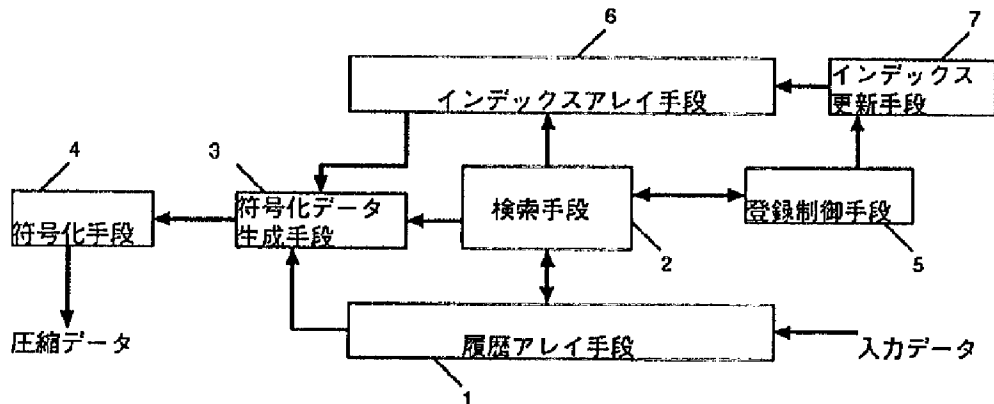
【図13】トライを用いた方法を示す図である。

【図14】ハッシュテーブルに衝突用の線形リストを備  
えた方法を示す図である。

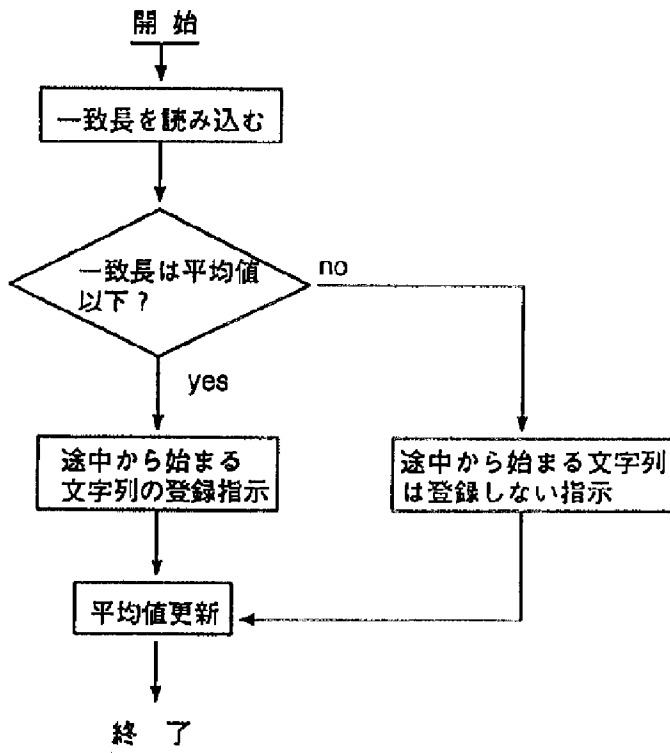
#### 【符号の説明】

- 1 履歴アレイ手段
- 2 検索手段
- 3 符号化データ生成手段
- 4 符号化手段
- 5 登録制御手段
- 6 インデックスアレイ手段
- 7 インデックス更新手段

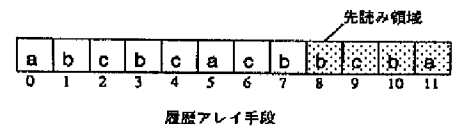
【図1】



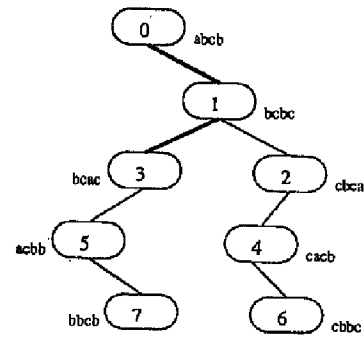
【図3】



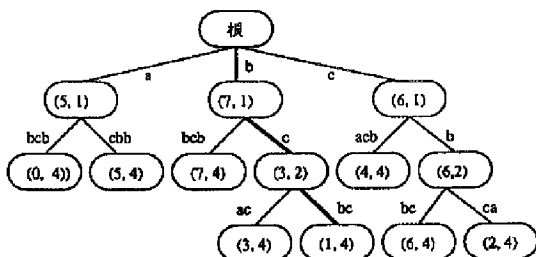
【図11】



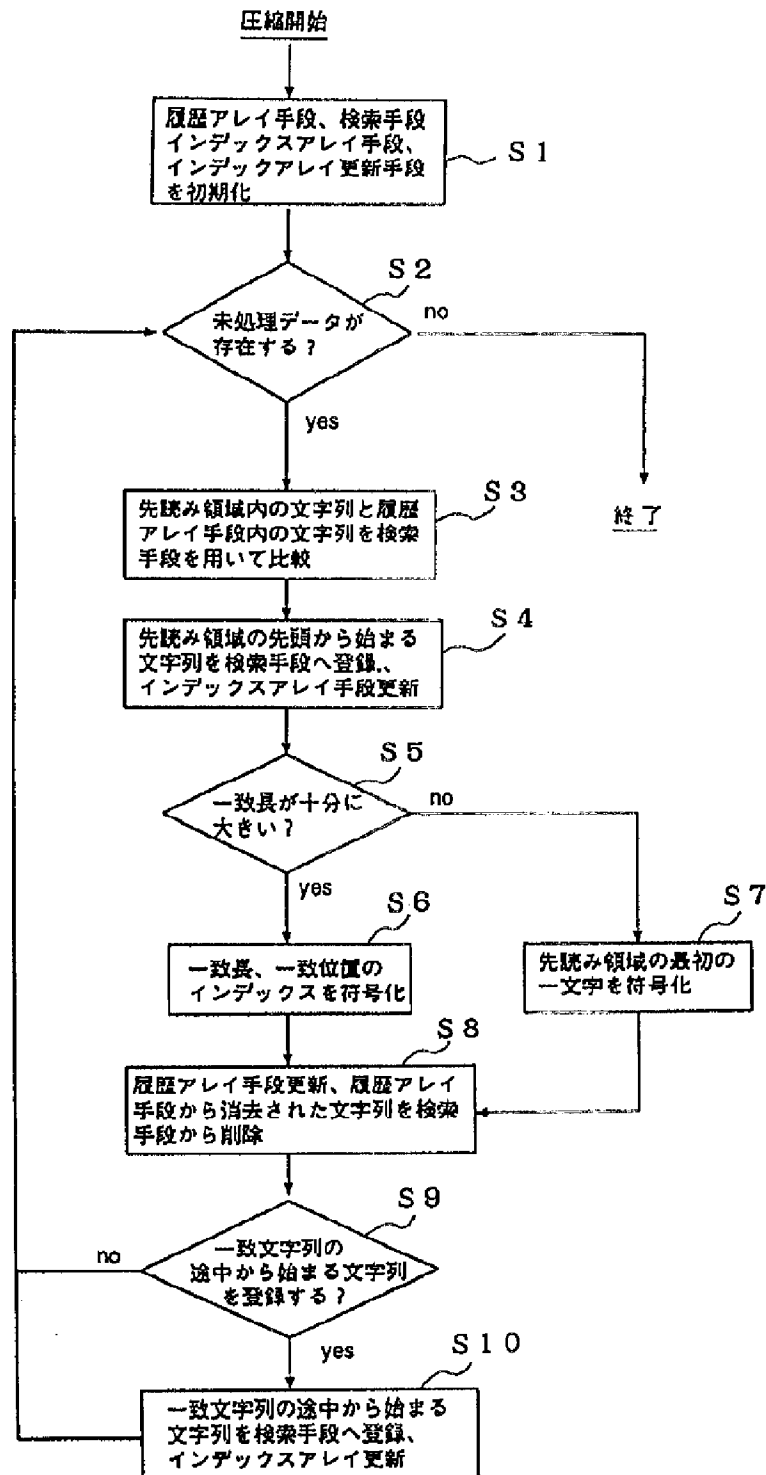
【図12】



【図13】

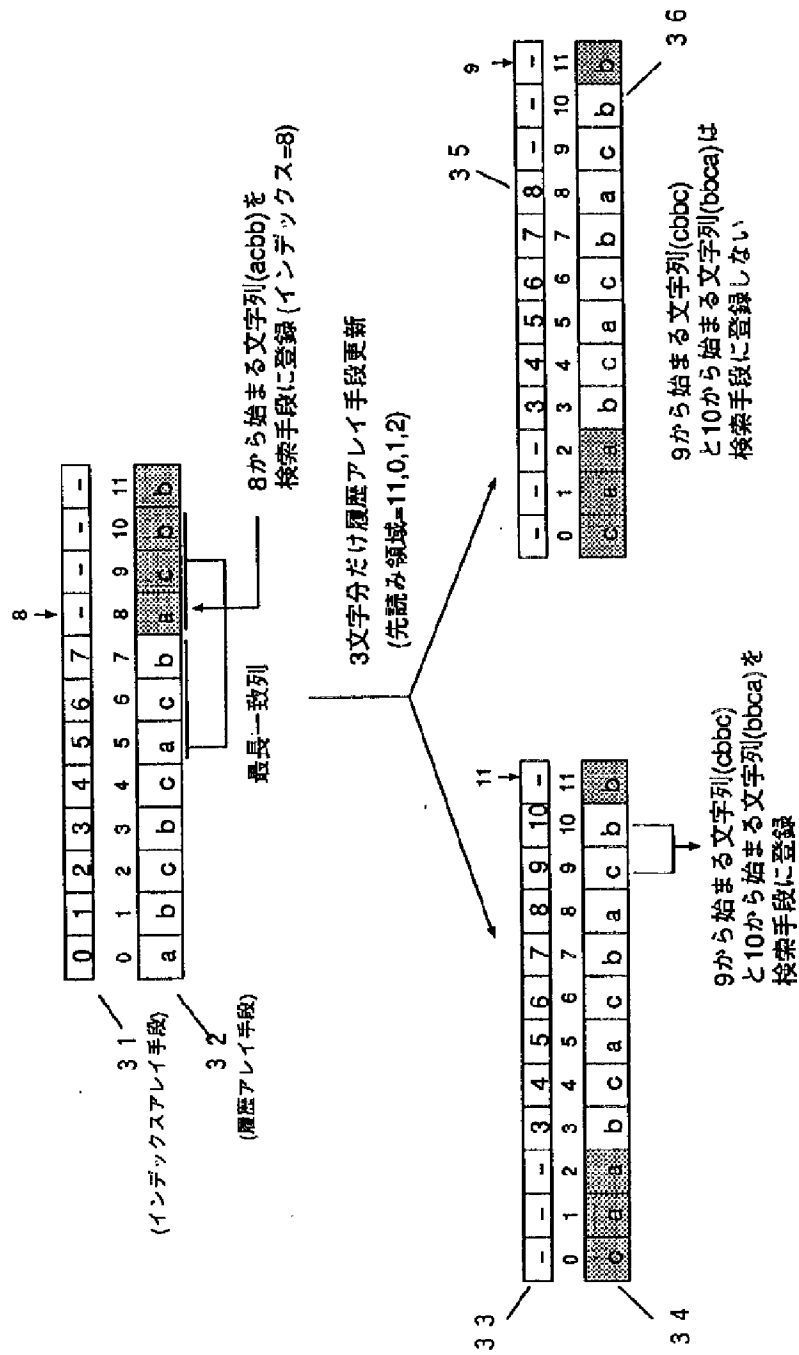


【図2】





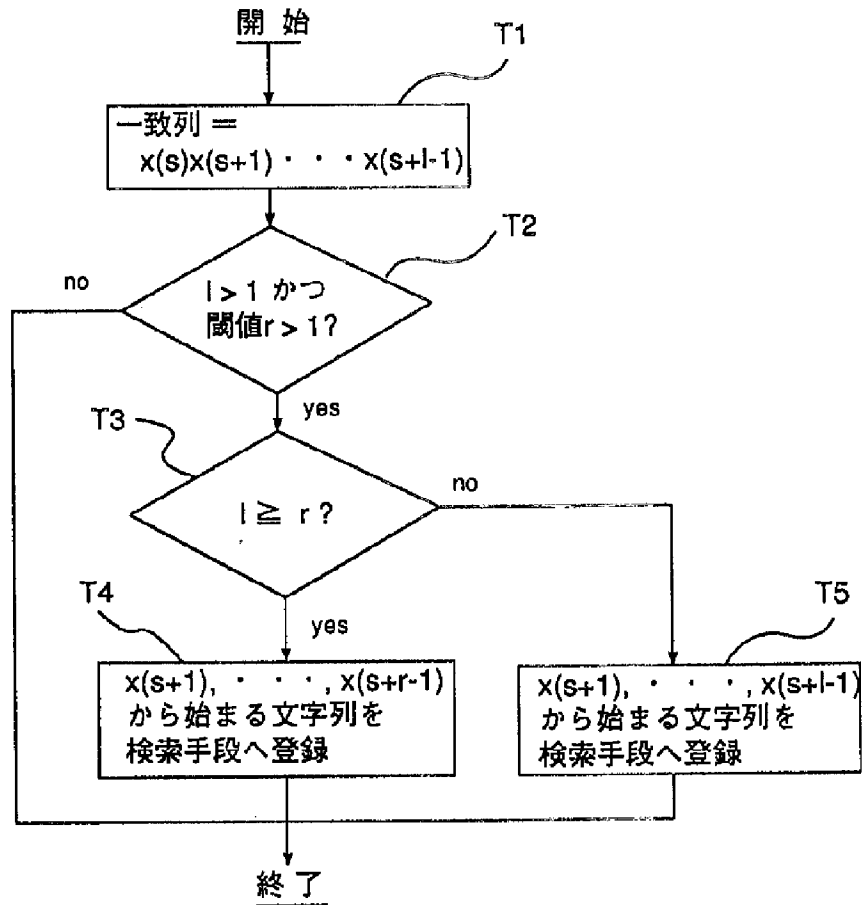
【図5】



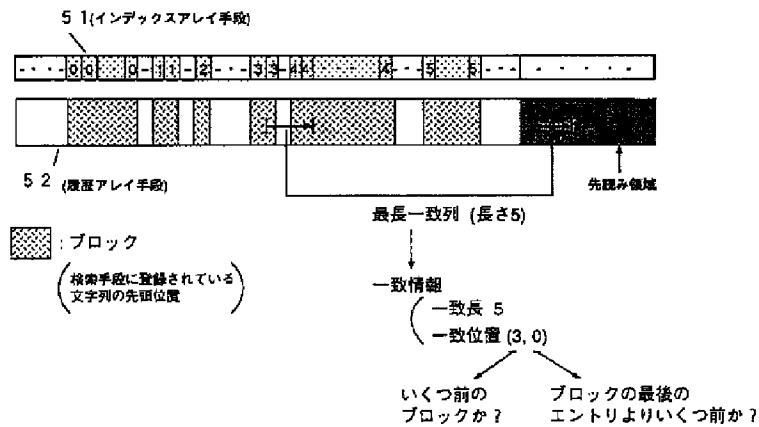
(b) 本発明で一致位置を基準とした場合で  
閾値を2としたときの登録制御

(a) 従来方式 または 本発明で一致位置を基準とした  
場合で閾値を4としたときの登録制御

【図6】

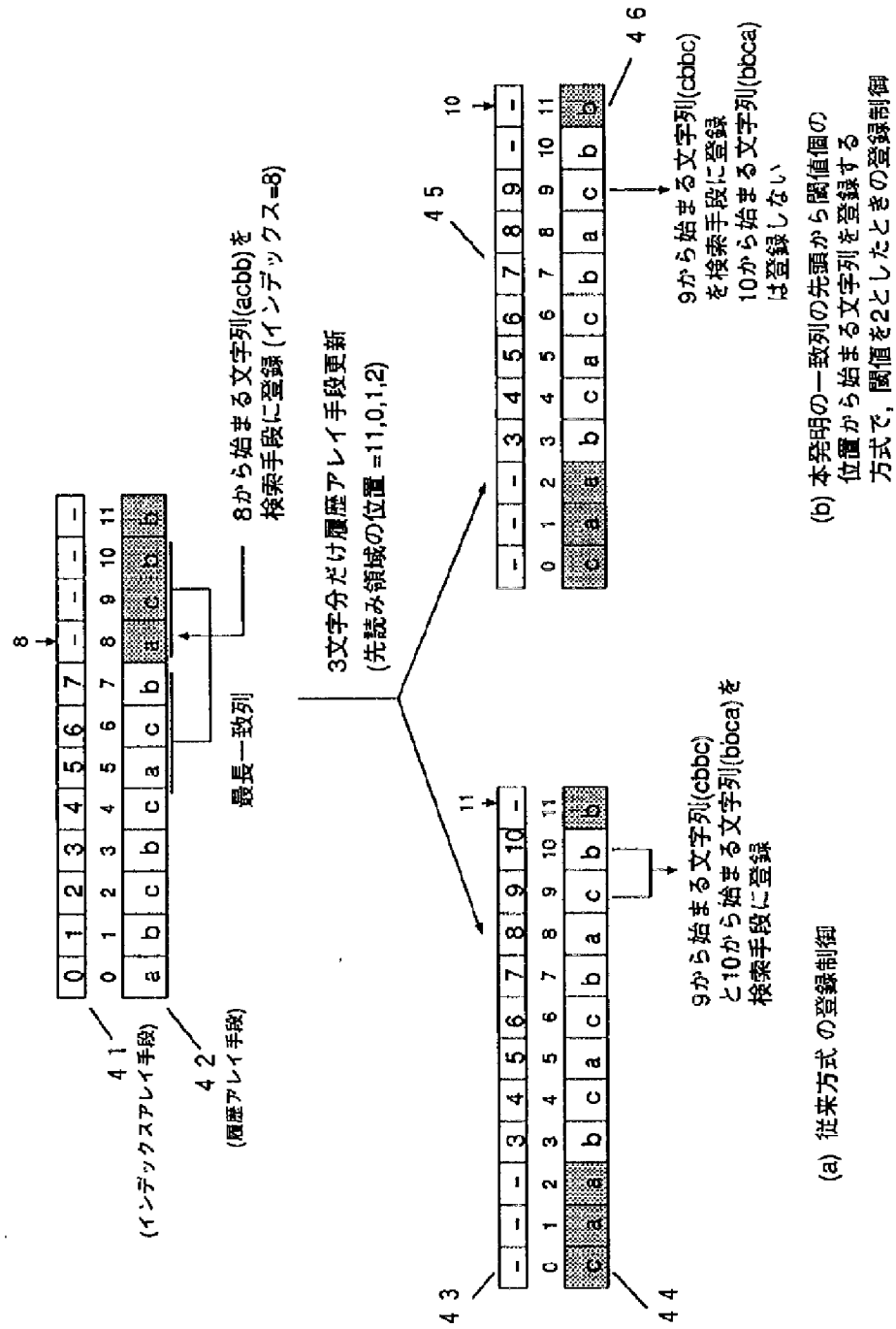


【図9】

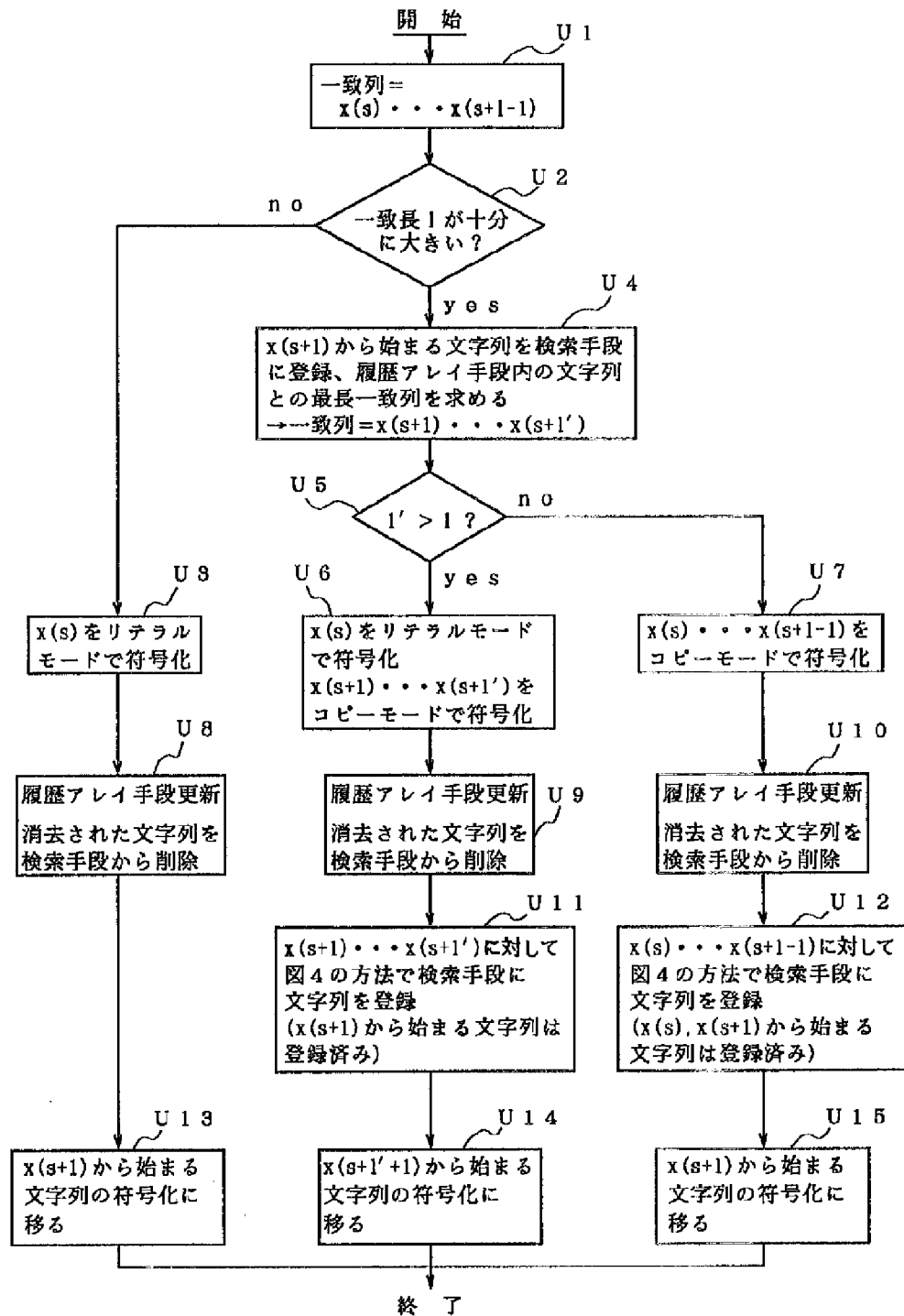




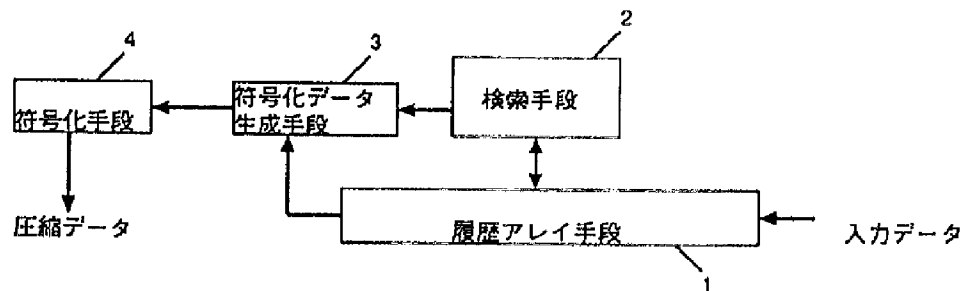
【図7】



【図8】



【図10】



【図14】

